Gradient-based training of probabilistic neural network with class-feature smoothing

Anonymous authors
Paper under double-blind review

Abstract

This paper proposes a gradient-based training procedure for probabilistic neural network with class- and feature-dependent smoothing parameters. The parameters are updated using analytically derived gradients defined for the normalized class outputs. Three initialization strategies are considered: class-related feature variance, median pairwise distance, and interquartile range. The method is evaluated on 20 classification datasets using repeated stratified 10-fold cross-validation. The variance-based initialization obtains the best mean rank, although the relative performance of the initialization methods is dataset-dependent. Statistical tests do not indicate significant differences between the considered initialization strategies.

1 Introduction

The probabilistic neural network (PNN) is a feedforward implementation of Parzen-window density estimation combined with the Bayes decision rule (Specht, 1990). Its architecture consists of input, pattern, summation, and decision layers. A key element of the PNN is the smoothing parameter, which controls the kernel width and directly affects the estimated class-conditional probability densities.

In this work, we propose a gradient-based training procedure in which separate smoothing parameters are assigned to individual class-feature pairs and adapted during learning. Their updates are obtained analytically from class-specific errors defined for the normalized PNN outputs. Three data-dependent initialization strategies are considered: class-related feature variance, median pairwise distance, and interquartile range. The method is evaluated on 20 classification datasets using repeated stratified 10-fold cross-validation. The analysis focuses on classification performance, learning behavior, and the influence of smoothing-parameter initialization.

The remainder of the paper is organized as follows. In Section 2, recent studies on PNN training are reviewed. Section 3 presents the PNN formulation, training procedure, and initialization strategies. Section 4 describes the datasets, Section 5 presents the experimental results, and Section 6 concludes the paper.

2 Literature review

Since the original formulation of the PNN by Specht (1990), numerous studies have addressed automatic selection of smoothing parameters and alternative PNN training procedures. Streit & Luginbuhl (1994) proposed maximum-likelihood training for Gaussian-kernel PNNs, whereas Yang & Chen (1998) extended this approach to heteroscedastic models and introduced Jackknife-based stabilization. Chtioui et al. (1998) optimized smoothing parameters using a two-stage procedure

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

combining conjugate-gradient and approximate Newton methods. Their results demonstrated the potential of gradient-based PNN optimization, although at a substantial computational cost.

Other studies focused on structural adaptation and non-gradient optimization. Mao et al. (2000) jointly optimized the smoothing parameter and selected representative pattern neurons using a genetic algorithm and forward-regression procedure. Rutkowski (2004) introduced adaptive PNNs for nonstationary classification, allowing probability-density estimates and decision surfaces to evolve over time. Georgiou et al. (2006) proposed self-adaptive PNNs in which global or feature-specific spread parameters were determined using golden-section search or particle swarm optimization. Zhong et al. (2007) introduced a computationally inexpensive gap-based smoothing-parameter estimator based on distances between neighboring samples.

The self-adaptive formulation was further generalized by Georgiou et al. (2008), who considered separate spread parameters for individual features and classes and optimized them using particle swarm optimization. Metaheuristic approaches were also investigated by Alweshah & Abdullah (2015), who applied firefly-based optimization to PNN weights, and by Naik et al. (2020), who optimized smoothing parameters using bat, grey wolf, and whale optimization algorithms and additionally considered Laplace kernels.

Kusy & Zajdel (2015) proposed reinforcement-learning procedures for adapting global, feature-specific, and class-feature-specific PNN smoothing parameters using $Q(0)$-learning, $Q(\lambda)$-learning, and stateless Q-learning. More recently, Kowalski et al. (2025) introduced a constrained hybrid metaheuristic combining several population-based and local-search optimization mechanisms for PNN learning.

Overall, previous studies have used maximum-likelihood estimation, gradient-based optimization, reinforcement learning, and metaheuristics to determine PNN smoothing parameters. Some methods allow feature-specific or class-feature-specific parameterization. Among them, Chtioui et al. (1998) is the closest to the present study due to its gradient-based formulation. In contrast, the proposed method updates a class- and feature-specific smoothing matrix $\mathbf{\Sigma}$ using analytically derived gradients and mini-batch stochastic gradient descent.

3 PNN formulation and training procedure

A probabilistic neural network consists of input, pattern, summation, and output layers. Pattern neurons correspond to training samples and evaluate kernel responses for the input vector $\mathbf{x}$. The summation layer aggregates these responses separately for each class, while the output layer assigns $\mathbf{x}$ to the class with the highest normalized response.

For class j, the summation-layer output is determined by class-conditional kernel density estimator:

$$f_j(\mathbf{x}) = \frac{1}{L_j (2\pi)^{I/2} \det \mathbf{\Sigma}_j} \sum_{l=1}^{L_j} \exp\left(-\frac{1}{2} \left(\mathbf{x} - \mathbf{x}_l^{(j)} \right)^T \mathbf{\Sigma}_j^{-2} \left(\mathbf{x} - \mathbf{x}_l^{(j)} \right) \right), \tag{1}$$

where L_j denotes the number of training samples in class j, $\mathbf{\Sigma}_j = \operatorname{diag}(\sigma_{j1}, \sigma_{j2}, \ldots, \sigma_{jI})$ and $\det \mathbf{\Sigma}_j = \prod_{i=1}^{I} \sigma_{ji}$.

The normalized output corresponding to class j is:

$$g_j(\mathbf{x}) = \frac{f_j(\mathbf{x})}{\sum_{k=1}^{J} f_k(\mathbf{x})}. \tag{2}$$

The predicted class is determined by the maximum value of g_j.

3.1 Gradient-based learning of the smoothing parameters

The proposed learning procedure adapts the class- and feature-dependent smoothing parameters σ_{ji}. For each normalized output g_j, a class-specific quadratic error is defined as:

$$E_j = \frac{1}{2}\left(t_j - g_j\right)^2, \tag{3}$$

where t_j is the j-th component of the one-hot encoded target vector. The parameter σ_{ji} is updated according to the gradient of the corresponding error E_j. The resulting sample-wise gradient is:

$$\frac{\partial E_j}{\partial \sigma_{ji}} = -\frac{1}{\sum_{k=1}^{J} f_k}(t_j - g_j)(1 - g_j)\frac{\partial f_j}{\partial \sigma_{ji}}. \tag{4}$$

During training, the sample-wise gradients are averaged over each mini-batch and used to update the smoothing parameters by gradient descent. The complete derivation and update procedure are given in Appendix A.

3.2 Initialization of the smoothing parameters

Gradient-based learning requires an initial smoothing matrix $\mathbf{\Sigma} = [\sigma_{ji}]$. Since each parameter is associated with a specific class and input feature, the initialization is performed separately for each class–feature pair. This allows the initial smoothing scales to reflect the corresponding characteristics of the training data before the gradient-based adaptation begins.

Three data-dependent initialization strategies are considered: class-related feature-based variance (V), median of class-related feature pairwise distances (MP), and class-related interquartile range (Q). The motivation for the class- and feature-dependent parameterization is discussed below, followed by the definitions of the three initialization procedures.

3.3 Motivation for class- and feature-dependent smoothing parameters

The use of class- and feature-dependent smoothing parameters can be justified from the form of the Gaussian kernel employed in the PNN. For an input pattern $\boldsymbol{x}$ and a training pattern $\boldsymbol{x}_l^{(j)}$ belonging to class j, the kernel response is determined by: $\exp\left(-\sum_{i=1}^{I}\frac{\left(x_i - x_{li}^{(j)}\right)^2}{2\sigma_{ji}^2}\right)$. Therefore, the matrix $\mathbf{\Sigma}$ defines a class-dependent diagonal scaling of the feature space. A larger value of σ_{ji} reduces the influence of the difference in feature i for class j, whereas a smaller value increases its contribution to the kernel response. Thus, the smoothing parameters determine the local sensitivity of the classifier to deviations in individual features within individual classes. This formulation is particularly justified when the class-conditional feature distributions exhibit heterogeneous dispersions.

Now let the class-conditional covariance structure be approximated by a diagonal matrix:

$$\mathbf{V}^{(j)} = \operatorname{diag}\left(v_{j1}, v_{j2}, \ldots, v_{jI}\right), \tag{5}$$

where v_{ji} denotes the within-class variance of feature i in class j. Under this assumption, the smoothing scale associated with feature i and class j should be related to the corresponding within-class dispersion:

$$\sigma_{ji}^2 \propto v_{ji}. \tag{6}$$

Thus, features with high within-class variability should be associated with larger smoothing parameters; features characterized by concentrated class-conditional distributions should receive smaller values. Moreover, the same feature may require different smoothing scales in different classes.

The classical PNN with a single global smoothing parameter represents a restricted special case of the proposed formulation:

$$\sigma_{ji} = \sigma, \qquad \forall i \in \{1, \ldots, I\}, \quad \forall j \in \{1, \ldots, J\}. \tag{7}$$

Let $\mathcal{S}_{\text{global}}$ denote the set of smoothing-parameter matrices satisfying equation 7:

$$\mathcal{S}_{\text{global}} = \{\boldsymbol{\Sigma} \mid \sigma_{ji} = \sigma > 0, \quad \forall j \in \{1, \ldots, J\}, \quad \forall i \in \{1, \ldots, I\}\}. \tag{8}$$

Let $\mathcal{S}_{\text{CF}}$ denote the set of all positive class- and feature-dependent smoothing-parameter matrices:

$$\mathcal{S}_{\text{CF}} = \{\boldsymbol{\Sigma} \mid \sigma_{ji} > 0, \quad \forall j \in \{1, \ldots, J\}, \quad \forall i \in \{1, \ldots, I\}\}. \tag{9}$$

Then $\mathcal{S}_{\text{global}} \subsetneq \mathcal{S}_{\text{CF}}$. Hence, the class- and feature-dependent formulation includes the global PNN as a particular case while providing a strictly richer parameter space. For a given training criterion this implies:

$$\min_{\boldsymbol{\Sigma} \in \mathcal{S}_{\text{CF}}} E(\boldsymbol{\Sigma}) \leq \min_{\boldsymbol{\Sigma} \in \mathcal{S}_{\text{global}}} E(\boldsymbol{\Sigma}). \tag{10}$$

where $E(\boldsymbol{\Sigma})$ is training objective function defined over the corresponding parameter space. Equation equation 10 does not guarantee improved generalization for every dataset. It shows, however, that the proposed parameterization does not impose the restrictive assumption that all features and all classes should share an identical smoothing scale. The considered initialization strategies therefore provide data-dependent estimates of the class- and feature-specific local scales before the subsequent gradient-based optimization of $\boldsymbol{\Sigma}$.

In this work, we consider three initialization strategies for the smoothing parameter matrix. Their purpose is to provide a suitable starting point for the subsequent optimization of $\boldsymbol{\Sigma}$. The considered solutions include the class-related approaches: feature-based variance, the median of feature pairwise distances and the minimum centroid-based distance. These strategies are briefly characterized in the following subsections.

3.3.1 Class-related feature-based variance

In this approach, the single smoothing parameter is initialized as follows:

$$\sigma_{ji} = \text{var}\left(\boldsymbol{x}_i^{(j)}\right), \tag{11}$$

where $\boldsymbol{x}_i^{(j)} = [x_{1i}^{(j)}, x_{2i}^{(j)}, \ldots, x_{L_j i}^{(j)}]$ is the i-feature column vector that belongs to class j. The parameter σ_{ji} is computed for each class and feature separately.

3.3.2 Median of class-related feature pairwise distances

This solution is based on the median of the distances between records of the same class. In particular, for each class j and feature i one computes:

$$d_{kl,i}^{(j)} = \left| x_{ki}^{(j)} - x_{li}^{(j)} \right|, \quad k < l. \tag{12}$$

The constraint in (12) allows for computing the distance between the same records only once; e.g., for $L_j = 4$, we determine 6 distances: $\left\{d_{12,i}^{(j)}, d_{13,i}^{(j)}, d_{14,i}^{(j)}, d_{23,i}^{(j)}, d_{24,i}^{(j)}, d_{34,i}^{(j)}\right\}$. The smoothing parameter is thereafter initialized as follows:

$$\sigma_{ji} = \text{median}\left\{d_{kl,i}^{(j)}\right\} \quad \text{s.t.} \quad 1 \leq k < l \leq L_j. \tag{13}$$

This method is more robust to outliers than the method based directly on variance.

3.3.3 Class-related interquartile range

In this initialization procedure, the class-related first and third empirical quartiles are first determined for the feature vector $\boldsymbol{x}_i^{(j)} = [x_{1i}^{(j)}, x_{2i}^{(j)}, \ldots, x_{L_j i}^{(j)}]$,. The quartiles are defined as:

$$Q_{0.25,i}^{(j)} = \text{quartile}_{0.25}\left(\boldsymbol{x}_i^{(j)}\right), \quad Q_{0.75,i}^{(j)} = \text{quartile}_{0.75}\left(\boldsymbol{x}_i^{(j)}\right). \tag{14}$$

The smoothing parameter associated with class j and feature i is then initialized as:

$$\sigma_i^{(j)} = Q_{0.75,i}^{(j)} - Q_{0.25,i}^{(j)} = \text{IQR}\left(\boldsymbol{x}_i^{(j)}\right). \tag{15}$$

Thus, $\sigma_i^{(j)}$ corresponds to the width of the central 50% of the class-conditional feature distribution. Compared with variance-based initialization, this estimate is less sensitive to extreme observations and provides a robust measure of feature dispersion within each class.

4 Datasets

The proposed PNN training algorithm is evaluated on 20 benchmark classification datasets from the UCI Machine Learning Repository. To provide diverse experimental conditions, the datasets cover sample sizes from 150 to 4601 records, feature-space dimensionalities from 4 to 100, and $\{2, 3, 5, 6, 10\}$ decision classes. This diversity is particularly relevant to the proposed PNN, since the smoothing parameters are represented by a class-feature matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{J \times I}$, while the computational cost of the standard PNN also depends directly on the number of training samples.

For compactness, each dataset is characterized below by (N, I, J), denoting the number of records, input features, and decision classes, respectively: Brain tumor (BT; $3762, 13, 2$), Chess (CH; $3196, 36, 2$), Cardiotocography (CTG; $2126, 22, 3$), Dermatology (D; $358, 34, 6$), Diagnostic breast cancer (DBC; $569, 30, 2$), Ecoli (E; $327, 6, 5$), Hill valley (HV; $606, 100, 2$), Ionosphere (IO; $352, 33, 2$), Iris (IR; $150, 4, 3$), Liver (L; $345, 6, 2$), Optical digits (OD; $3823, 62, 10$), Prognostic Wisconsin Breast Cancer (PBC; $198, 32, 2$), Pima Indians diabetes (PID; $768, 8, 2$), QSAR biodegradation (QSAR; $1055, 41, 2$), Sports articles (SA; $1000, 57, 2$), Spam base (SB; $4601, 57, 2$), Seeds (SE; $210, 7, 3$), Sonar (SO; $208, 60, 2$), Spectf (SP; $267, 44, 2$), and Statlog (ST; $4435, 36, 6$).

5 Results

This section presents the experimental results obtained for the proposed gradient-trained PNN under the considered initialization methods of the smoothing-parameter matrix $\boldsymbol{\Sigma}$. Unless otherwise stated, the learning rate was fixed at $\eta = 0.05$, whereas the mini-batch size was set to 32. The evaluation was performed using repeated stratified K-fold cross-validation with $K = 10$, repeated $S = 10$ times for each dataset and experimental configuration.

The presented analysis addresses several complementary aspects of the proposed approach. First, the evaluation protocol used to aggregate the repeated cross-validation outcomes is formally defined. Next, representative learning dynamics and the distributions of the minimum cross-validation errors are visualized. The results are then summarized across all datasets using aggregate performance measures, method rankings, and statistical tests. Finally, the computational runtime of the considered initialization strategies is analyzed.

5.1 Evaluation approach

Let $s \in \{1, \ldots, S\}$ denote the repetition index and $k \in \{1, \ldots, K\}$ denote the fold index within repetition s. For each pair (s, k), the PNN model is trained on the corresponding training partition and evaluated after every epoch e on the held-out fold.

Let $E_{\mathrm{CV}}^{(s,k)}(e)$ denote the cross-validation error obtained for repetition s, fold k, and epoch e. For each repetition, the epoch-wise cross-validation trajectory is first averaged over the K folds:

$$\overline{E}_{\mathrm{CV}}^{(s)}(e) = \frac{1}{K} \sum_{k=1}^{K} E_{\mathrm{CV}}^{(s,k)}(e). \tag{16}$$

The final cross-validation error trajectory is then obtained by averaging the repetition-wise trajectories over all S repetitions:

$$\overline{E}_{\mathrm{CV}}(e) = \frac{1}{S} \sum_{s=1}^{S} \overline{E}_{\mathrm{CV}}^{(s)}(e). \tag{17}$$

Additionally, for every fold, the epoch associated with the minimum cross-validation error is:

$$e_{\min}^{(s,k)} = \arg\min_{e} E_{\mathrm{CV}}^{(s,k)}(e). \tag{18}$$

Based on (18), the corresponding minimum cross-validation error is defined:

$$E_{\min}^{(s,k)} = E_{\mathrm{CV}}^{(s,k)} \left(e_{\min}^{(s,k)} \right). \tag{19}$$

To obtain one summary measure for each repeated K-fold cross-validation run, the fold-wise minimum errors are averaged within repetition s:

$$\overline{E}_{\min}^{(s)} = \frac{1}{K} \sum_{k=1}^{K} E_{\min}^{(s,k)}. \tag{20}$$

This protocol characterizes both the average cross-validation learning dynamics and the variability of the minimum cross-validation errors obtained across repeated cross-validation runs.

5.2 Presentation of cross-validation results

Figure 1 illustrates the mean cross-validation error trajectories $\overline{E}_{\mathrm{CV}}(e)$ across epochs in the left panels and the distributions of the S repetition-wise mean minimum cross-validation errors $\overline{E}_{\min}^{(s)}$ in the right panels. Due to the large number of experiments, representative PNN results are shown for three selected datasets illustrating distinct behaviors: (i) the lowest-error case: IR, (ii) an initialization-sensitive case: CH, and (iii) the highest-error case: HV.

The first example, the low-error IR dataset, shows an initial decrease of the mean cross-validation error for MP and Q initialization method, followed by an increase in later epochs. In contrast, V initialization strategy remains more stable during training. The distribution plots indicate that Q obtains the lowest $\overline{E}_{\min}$, followed by MP, whereas V yields the highest error and larger variability.

The CH dataset illustrates an initialization-sensitive case. Here, V clearly outperforms the remaining initialization strategies: PNN error decreases rapidly and remains much lower than those

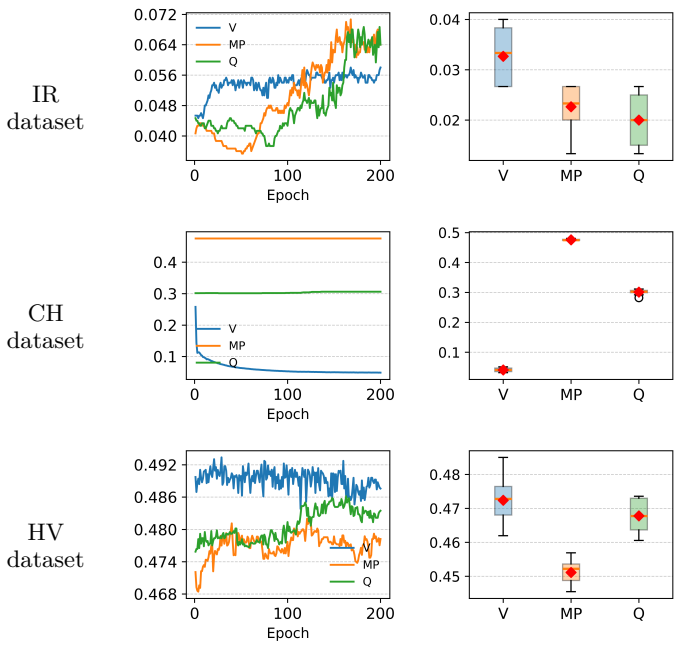


Figure 1: Cross-validation error trajectories (left) and repetition-wise minimum-error distributions (right) for IR, CH, and HV datasets. Results are shown for V, MP, and Q initialization; red diamonds indicate mean $\overline{E}_{\min}^{(s)}$.

obtained when $\boldsymbol{\Sigma}$ is initialized with MP and Q methods. The boxplots support this observation, showing the lowest $\overline{E}_{\min}$ and very small variability for V, while MP gives the highest error and Q remains intermediate.

The HV dataset represents a difficult classification case. All methods remain at relatively high CV error levels, but MP initialization method consistently obtains the lowest errors, followed by Q. In contrast, V produces the highest errors and the largest spread.

These examples are intended to illustrate representative learning behaviors, whereas the complete quantitative comparison across all datasets is reported below in Table 1. The corresponding error trajectories and distribution plots for all datasets are provided in Appendix B section.

Table 1 reports the mean minimum cross-validation error $\overline{E}_{\min}$ of the PNN, with the corresponding standard deviation, for the considered V, MP, and Q smoothing-parameter initialization strategies for all datasets. The best initialization method for each dataset is indicated in the last column.

The results show that the initialization strategies exhibit dataset-dependent performance patterns. Therefore, in addition to the number of wins, the mean rank is computed across datasets. For each dataset, rank 1 is assigned to the method yielding the lowest mean cross-validation error. The mean rank of method m is defined as:

$$\overline{r}_m = \frac{1}{D} \sum_{d=1}^{D} r_{d,m}, \tag{21}$$

where D denotes the number of datasets and $r_{d,m}$ is the rank of method m on dataset d. This ranking-based summary is used as the primary aggregate comparison, since it is less affected by differences in the absolute error scale across datasets.

As shown, the variance-based initialization achieves the lowest mean cross-validation error on 9 datasets and obtains the best mean rank of 1.85. The Q initialization achieves the lowest error on 7 datasets, with a mean rank of 2.00, whereas the median-pairwise initialization is the best method on 4 datasets and obtains a mean rank of 2.15. These results indicate that no single initialization strategy dominates across all datasets. Nevertheless, V and Q approaches provide the best results more frequently than the MP method, with the variance-based initialization showing the strongest aggregate performance according to the mean-rank criterion.

Table 1: Mean cross-validation error and standard deviation obtained for the three smoothing-parameter initialization methods. The best result for each dataset is highlighted in bold.

Dataset	V	MP	Q	Best
BT	0.031 ± 0.009	0.031 ± 0.018	$\mathbf{0.015 \pm 0.001}$	Q
CH	$\mathbf{0.041 \pm 0.007}$	0.476 ± 0.003	0.301 ± 0.008	V
CTG	$\mathbf{0.035 \pm 0.006}$	0.045 ± 0.008	0.051 ± 0.008	V
D	$\mathbf{0.082 \pm 0.010}$	0.201 ± 0.011	0.307 ± 0.022	V
DBC	0.141 ± 0.051	$\mathbf{0.035 \pm 0.004}$	0.060 ± 0.007	MP
E	0.188 ± 0.015	0.188 ± 0.013	$\mathbf{0.185 \pm 0.013}$	Q
HV	0.472 ± 0.007	$\mathbf{0.451 \pm 0.004}$	0.468 ± 0.005	MP
IO	0.043 ± 0.005	$\mathbf{0.027 \pm 0.003}$	0.071 ± 0.007	MP
IR	0.033 ± 0.006	0.023 ± 0.005	$\mathbf{0.020 \pm 0.005}$	Q
L	0.310 ± 0.008	0.279 ± 0.010	$\mathbf{0.265 \pm 0.013}$	Q
OD	$\mathbf{0.076 \pm 0.004}$	0.287 ± 0.014	0.198 ± 0.016	V
PBC	$\mathbf{0.193 \pm 0.011}$	0.262 ± 0.018	0.242 ± 0.015	V
PID	0.213 ± 0.006	0.222 ± 0.006	$\mathbf{0.209 \pm 0.003}$	Q
QSAR	$\mathbf{0.147 \pm 0.009}$	0.198 ± 0.031	0.212 ± 0.026	V
SA	$\mathbf{0.168 \pm 0.003}$	0.206 ± 0.015	0.221 ± 0.034	V
SB	$\mathbf{0.098 \pm 0.006}$	0.229 ± 0.002	0.196 ± 0.020	V
SE	0.057 ± 0.006	0.042 ± 0.005	$\mathbf{0.039 \pm 0.005}$	Q
SO	0.186 ± 0.011	0.181 ± 0.011	$\mathbf{0.121 \pm 0.008}$	Q
SP	$\mathbf{0.176 \pm 0.007}$	0.200 ± 0.015	0.210 ± 0.015	V
ST	0.110 ± 0.001	$\mathbf{0.097 \pm 0.002}$	0.111 ± 0.003	MP
Wins	**9**	4	7	–
Mean rank	**1.85**	2.15	2.00	–

To complement the method-wise comparison, we additionally examine the dataset-level behavior of the PNN independently of a particular initialization strategy. For each dataset d, the aggregate error is computed across the $M = 3$ initialization methods as:

$$\overline{E}_{\text{agg}}^{(d)} = \frac{1}{M} \sum_{m=1}^{M} \overline{E}_{\text{min}}^{(d,m)}, \tag{22}$$

where $\overline{E}_{\text{min}}^{(d,m)}$ denotes the mean minimum cross-validation error obtained for dataset d and initialization method m. The corresponding between-method standard deviation is determined as:

$$s_{\text{agg}}^{(d)} = \sqrt{\frac{1}{M-1} \sum_{m=1}^{M} \left(\overline{E}_{\text{min}}^{(d,m)} - \overline{E}_{\text{agg}}^{(d)} \right)^2}. \tag{23}$$

The lowest aggregate errors are obtained for the IR dataset ($\overline{E}^{(d)}_{\text{agg}} = 0.025$, $s^{(d)}_{\text{agg}} = 0.007$) and the BT dataset ($\overline{E}^{(d)}_{\text{agg}} = 0.026$, $s^{(d)}_{\text{agg}} = 0.009$), indicating that these datasets are classified accurately by the PNN almost independently of the initialization strategy. In contrast, the highest aggregate errors are observed for the HV dataset ($\overline{E}^{(d)}_{\text{agg}} = 0.464$, $s^{(d)}_{\text{agg}} = 0.011$) and the L dataset ($\overline{E}^{(d)}_{\text{agg}} = 0.285$, $s^{(d)}_{\text{agg}} = 0.023$). This suggests that these datasets are generally more difficult for the PNN.

The between-method standard deviation also provides information about the sensitivity of the model to the initialization of the smoothing-parameter matrix. The largest values are obtained for the CH dataset ($s^{(d)}_{\text{agg}} = 0.219$) and the D dataset ($s^{(d)}_{\text{agg}} = 0.113$), indicating that, for these datasets, the choice of initialization strategy has a substantial influence on the final cross-validation error.

5.3 Statistical analysis

The Friedman test is applied to the dataset-wise ranks of the three initialization strategies. The result $\chi^2_F = 0.90$, $p = 0.638$ does not indicate statistically significant differences across the datasets.

Pairwise Wilcoxon signed-rank tests with Holm correction likewise show no significant differences between V and MP ($p_{\text{Holm}} = 0.568$), V and Q ($p_{\text{Holm}} = 0.568$), or MP and Q ($p_{\text{Holm}} = 0.784$). Thus, the better mean rank and larger number of wins obtained by V should be interpreted as a descriptive result rather than evidence of statistically significant superiority.

5.4 Summary

The cross-validation trajectories and error distributions presented in Figure 1 and and Figures 3–7 in Appendix B show that the effect of smoothing-parameter initialization is strongly dataset-dependent. No initialization strategy consistently dominates across all datasets, although the variance-based method obtains the best mean rank and the highest number of wins.

The Friedman test and pairwise Wilcoxon tests do not indicate statistically significant differences between the considered initialization strategies. Thus, the observed differences should be interpreted as descriptive rather than statistically confirmed.

The comparison with published PNN results in Appendix C shows that the proposed training procedure achieves competitive errors on several common datasets. However, differences in preprocessing, data partitions, and evaluation protocols prevent direct claims of superiority over previously published methods.

6 Conclusions

This study introduced a gradient-based training procedure for probabilistic neural network with class- and feature-dependent smoothing parameters. The parameters are adapted with respect to class-specific errors defined for the normalized PNN outputs, while three data-dependent initialization strategies were considered. Experiments on 20 classification datasets showed that the effect of initialization is dataset-dependent. The variance-based initialization obtained the best mean rank and the highest number of wins, although the Friedman and pairwise Wilcoxon tests did not indicate statistically significant differences between the initialization strategies. The results therefore support gradient-based adaptation of class- and feature-dependent smoothing parameters, while indicating that the choice of their initialization remains problem-dependent.

References

Mohammed Alweshah and Salwani Abdullah. Hybridizing firefly algorithms with a probabilistic neural network for solving classification problems. *Applied Soft Computing*, 35:513–524, 2015. doi: 10.1016/j.asoc.2015.06.018.

Younes Chtioui, Suranjan Panigrahi, and Ronald A. Marsh. Conjugate gradient and approximate newton methods for an optimal probabilistic neural network for food color classification. *Optical Engineering*, 37(11):3015–3023, 1998. doi: 10.1117/1.601972.

Vasileios L. Georgiou, Nicos G. Pavlidis, Konstantinos E. Parsopoulos, Philipos D. Alevizos, and Michael N. Vrahatis. New self-adaptive probabilistic neural networks in bioinformatic and medical tasks. *International Journal on Artificial Intelligence Tools*, 15(3):371–396, 2006. doi: 10.1142/S0218213006002722.

Vasileios L. Georgiou, Philipos D. Alevizos, and Michael N. Vrahatis. Novel approaches to probabilistic neural networks through bagging and evolutionary estimating of prior probabilities. *Neural Processing Letters*, 27(2):153–162, 2008. doi: 10.1007/s11063-007-9066-5.

Piotr A. Kowalski, Szymon Kucharczyk, and Jacek Mandziuk. Constrained hybrid metaheuristic algorithm for probabilistic neural networks learning. *Information Sciences*, 713:122185, 2025. doi: 10.1016/j.ins.2025.122185.

Maciej Kusy and Roman Zajdel. Application of reinforcement learning algorithms for the adaptive computation of the smoothing parameter for probabilistic neural network. *IEEE Transactions on Neural Networks and Learning Systems*, 26(9):2163–2175, 2015. doi: 10.1109/TNNLS.2014.2376703.

K. Z. Mao, K.-C. Tan, and W. Ser. Probabilistic neural-network structure determination for pattern classification. *IEEE Transactions on Neural Networks*, 11(4):1009–1016, 2000. doi: 10.1109/72.857781.

Shraddha M. Naik, Ravi Prasad K. Jagannath, and Venkatanareshbabu Kuppili. Estimation of the smoothing parameter in probabilistic neural network using evolutionary algorithms. *Arabian Journal for Science and Engineering*, 45(4):2945–2955, 2020. doi: 10.1007/s13369-019-04227-5.

Leszek Rutkowski. Adaptive probabilistic neural networks for pattern classification in time-varying environment. *IEEE Transactions on Neural Networks*, 15(4):811–827, 2004. doi: 10.1109/TNN.2004.828757.

Donald F. Specht. Probabilistic neural networks. *Neural Networks*, 3(1):109–118, 1990. doi: 10.1016/0893-6080(90)90049-Q.

Roy L. Streit and Tod E. Luginbuhl. Maximum likelihood training of probabilistic neural networks. *IEEE Transactions on Neural Networks*, 5(5):764–783, 1994. doi: 10.1109/72.317728.

Zheng Rong Yang and Sheng Chen. Robust maximum likelihood training of heteroscedastic probabilistic neural networks. *Neural Networks*, 11(4):739–747, 1998. doi: 10.1016/S0893-6080(98)00024-0.

Mingyu Zhong, Dave Coggeshall, Ehsan Ghaneie, Thomas Pope, Mark Rivera, Michael Georgiopoulos, Georgios C. Anagnostopoulos, Mansooreh Mollaghasemi, and Samuel Richie. Gap-based estimation: Choosing the smoothing parameters for probabilistic and general regression neural networks. *Neural Computation*, 19(10):2840–2864, 2007. doi: 10.1162/neco.2007.19.10.2840.

A Derivation of the PNN smoothing-parameter update

The proposed training procedure associates the smoothing parameters σ_{ji} with class j and feature i. Consequently, the parameters $\mathbf{\Sigma}_j = (\sigma_{j1}, \ldots, \sigma_{jI})$ affect the class-conditional response f_j, whereas the responses f_k, $k \neq j$, do not directly depend on $\mathbf{\Sigma}_j$. Figure 2 illustrates the PNN architecture used in the proposed training procedure.

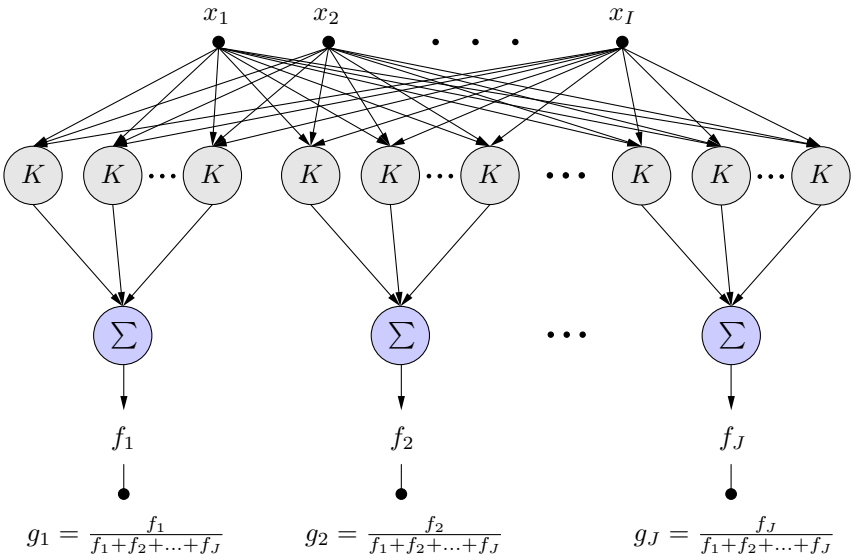


Figure 2: Probabilistic neural network with class- and feature-dependent smoothing parameters. The parameters associated with class j determine the corresponding class-conditional response f_j, which is subsequently normalized to obtain the output g_j.

The normalized output corresponding to class j is defined as:

$$g_j(\mathbf{x}) = \frac{f_j(\mathbf{x})}{\sum_{k=1}^{J} f_k(\mathbf{x})}. \tag{24}$$

For each output g_j, a class-specific quadratic error is defined as:

$$E_j = \frac{1}{2}\left(t_j - g_j\right)^2, \tag{25}$$

where t_j denotes the j-th component of the one-hot encoded target vector.

The derivative of E_j with respect to the smoothing parameter σ_{ji} is:

$$\frac{\partial E_j}{\partial \sigma_{ji}} = -\left(t_j - g_j\right)\frac{\partial g_j}{\partial \sigma_{ji}}. \tag{26}$$

Since σ_{ji} affects only the class-conditional response f_j, we have $\partial f_k / \partial \sigma_{ji} = 0$ for $k \neq j$. Consequently:

$$\frac{\partial}{\partial \sigma_{ji}}\left(\sum_{k=1}^{J} f_k\right) = \frac{\partial f_j}{\partial \sigma_{ji}}. \tag{27}$$

Applying the quotient rule to (24) gives:

$$\frac{\partial g_j}{\partial \sigma_{ji}} = \frac{\frac{\partial f_j}{\partial \sigma_{ji}} \sum_{k=1}^{J} f_k - f_j \frac{\partial f_j}{\partial \sigma_{ji}}}{\left(\sum_{k=1}^{J} f_k \right)^2} = \frac{\frac{\partial f_j}{\partial \sigma_{ji}}}{\sum_{k=1}^{J} f_k} \left(1 - \frac{f_j}{\sum_{k=1}^{J} f_k} \right). \tag{28}$$

Using (24), this becomes:

$$\frac{\partial g_j}{\partial \sigma_{ji}} = \frac{1}{\sum_{k=1}^{J} f_k} (1 - g_j) \frac{\partial f_j}{\partial \sigma_{ji}}. \tag{29}$$

Substitution into (26) yields:

$$\frac{\partial E_j}{\partial \sigma_{ji}} = -\frac{1}{\sum_{k=1}^{J} f_k} (t_j - g_j)(1 - g_j) \frac{\partial f_j}{\partial \sigma_{ji}}. \tag{30}$$

It remains to determine the derivative of the class-conditional response f_j with respect to σ_{ji}. According to (1), the response can be written as:

$$f_j(\mathbf{x}) = A_j \sum_{l=1}^{L_j} K_l^{(j)}(\mathbf{x}), \tag{31}$$

where

$$A_j = \frac{1}{L_j (2\pi)^{I/2} \prod_{q=1}^{I} \sigma_{jq}} \tag{32}$$

and:

$$K_l^{(j)}(\mathbf{x}) = \exp\left(-\sum_{q=1}^{I} \frac{\left(x_q - x_{lq}^{(j)} \right)^2}{2\sigma_{jq}^2} \right). \tag{33}$$

The derivative of the normalization factor A_j is:

$$\frac{\partial A_j}{\partial \sigma_{ji}} = -\frac{A_j}{\sigma_{ji}}. \tag{34}$$

For the kernel term:

$$\frac{\partial K_l^{(j)}}{\partial \sigma_{ji}} = K_l^{(j)} \frac{\left(x_i - x_{li}^{(j)} \right)^2}{\sigma_{ji}^3}. \tag{35}$$

Applying the product rule to (31) gives:

$$\frac{\partial f_j}{\partial \sigma_{ji}} = \frac{\partial A_j}{\partial \sigma_{ji}} \sum_{l=1}^{L_j} K_l^{(j)} + A_j \sum_{l=1}^{L_j} \frac{\partial K_l^{(j)}}{\partial \sigma_{ji}} = -\frac{f_j}{\sigma_{ji}} + A_j \sum_{l=1}^{L_j} \frac{\left(x_i - x_{li}^{(j)} \right)^2}{\sigma_{ji}^3} K_l^{(j)}. \tag{36}$$

After substituting (32) and (33), the derivative takes the explicit form:

$$\frac{\partial f_j}{\partial \sigma_{ji}} = -\frac{f_j}{\sigma_{ji}} + \frac{1}{L_j (2\pi)^{I/2} \prod_{q=1}^{I} \sigma_{jq}} \sum_{l=1}^{L_j} \frac{\left(x_i - x_{li}^{(j)}\right)^2}{\sigma_{ji}^3} \times \exp\left(-\sum_{q=1}^{I} \frac{\left(x_q - x_{lq}^{(j)}\right)^2}{2\sigma_{jq}^2} \right). \tag{37}$$

Equations (30) and (37) define the sample-wise gradient used to adapt each class- and feature-dependent smoothing parameter.

For a mini-batch $\mathcal{B}$, the gradients obtained for individual training samples are averaged:

$$\bar{\delta}_{ji}^{(\mathcal{B})} = \frac{1}{|\mathcal{B}|} \sum_{r \in \mathcal{B}} \frac{\partial E_{j,r}}{\partial \sigma_{ji}}, \tag{38}$$

where $E_{j,r}$ denotes the class-specific error associated with sample r. The smoothing parameter is then updated according to:

$$\sigma_{ji}^{(t+1)} = \sigma_{ji}^{(t)} - \eta \bar{\delta}_{ji}^{(\mathcal{B})}, \tag{39}$$

where η denotes the learning rate.

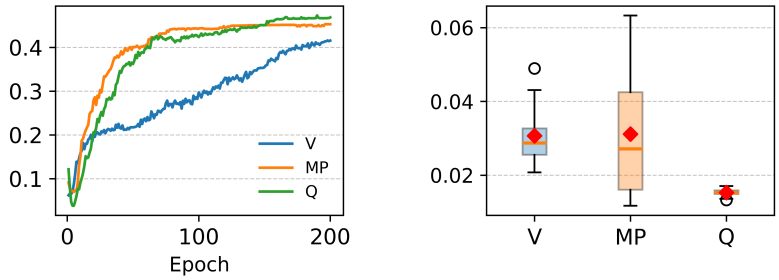
To preserve admissible smoothing parameters, the updated values are restricted to the positive interval $[\sigma_{\min}, \sigma_{\max}]$.

B PNN CLASSIFICATION RESULTS

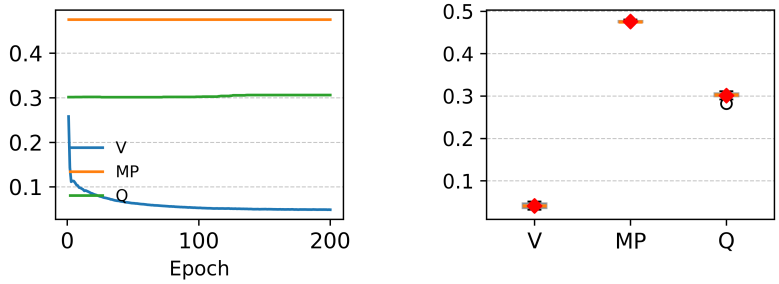
The following figures complement the representative results presented in the main text and provide the complete graphical comparison for all considered datasets. For each dataset, the left panel shows the mean cross-validation error trajectory across training epochs, whereas the right panel presents the distribution of the repetition-wise mean minimum cross-validation errors $\overline{E}^{(s)}_{\min}$. The results are reported for the V, MP, and Q smoothing-parameter initialization strategies and are grouped as follows:

- Figure 3: Brain tumor (BT), Chess (CH), Cardiotocography (CTG), and Dermatology (D) datasets;
- Figure 4: Diagnostic breast cancer (DBC), Ecoli (E), Hill valley (HV), and Ionosphere (IO) datasets;
- Figure 5: Iris (IR), Liver (L), Optical digits (OD) and Prognostic Wisconsin Breast Cancer (PBC);
- Figure 6: Pima Indians diabetes (PID), QSAR biodegradation (QSAR), Sports articles (SA), and Spam base (SB);
- Figure 7: Seeds (SB), Sonar (SO), Spectf (SP), and Statlog (ST)

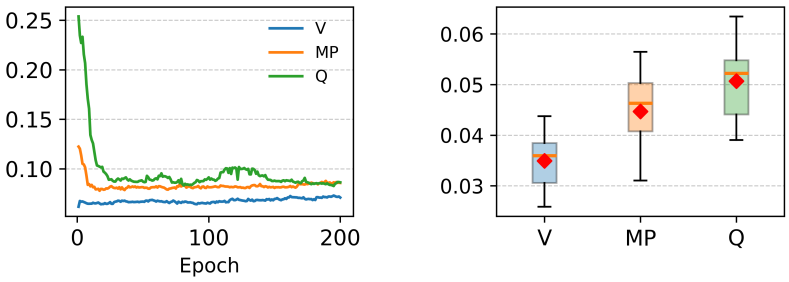
(a) CV errors for BT dataset



(b) CV errors for CH



(c) CV errors for CTG dataset



(a) CV errors for D dataset

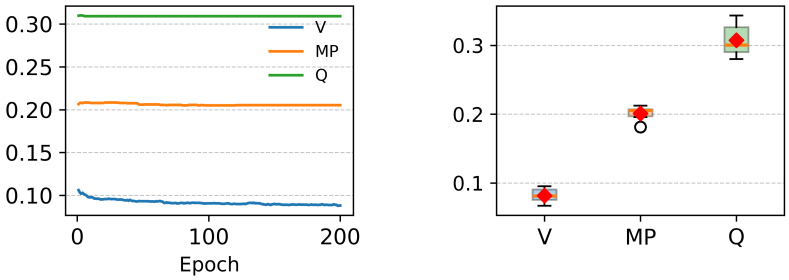
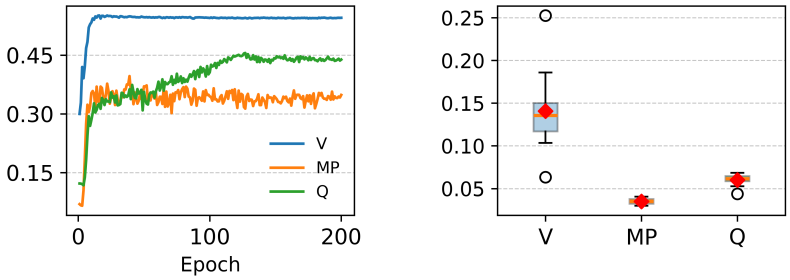
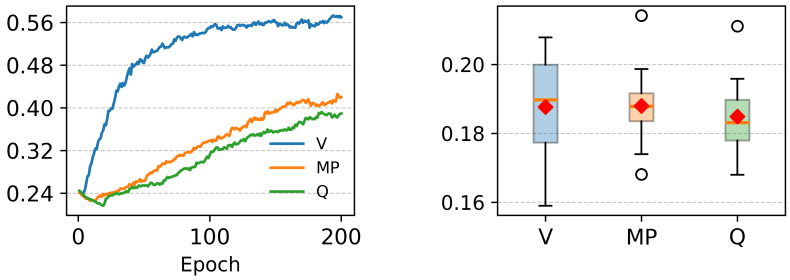


Figure 3: The trajectories of the mean cross-validation errors across epochs (left panels) and the distributions of mean minimum cross-validation errors (right panels) for the selected datasets In the boxplots, the $\overline{E}_{\min}$ errors are also included (marked using red diamond).

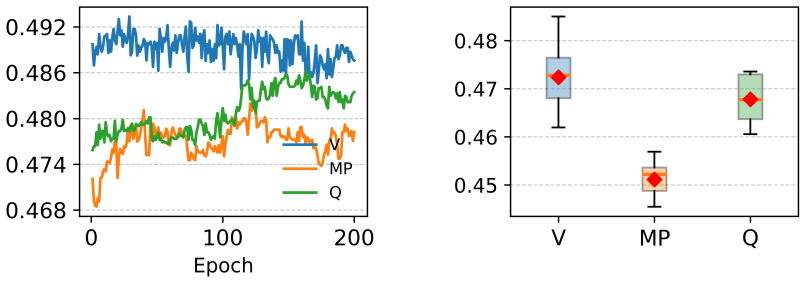
(c) CV errors for DBC dataset



(b) CV errors for E



(c) CV errors for HV dataset



(c) CV errors for IO dataset

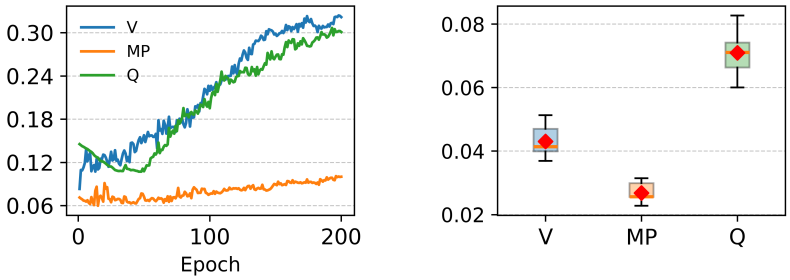
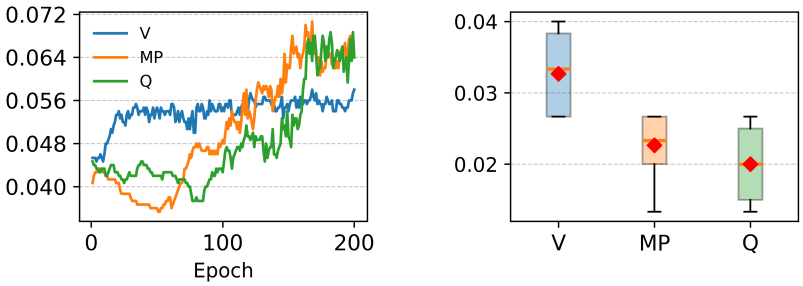
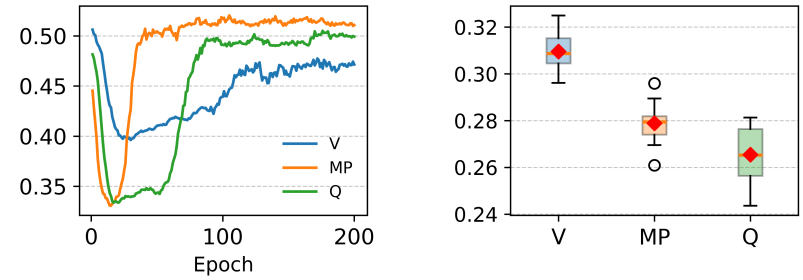


Figure 4: The trajectories of the mean cross-validation errors across epochs (left panels) and the distributions of mean minimum cross-validation errors (right panels) for the selected datasets In the boxplots, the $\overline{E}_{\min}$ errors are also included (marked using red diamond).

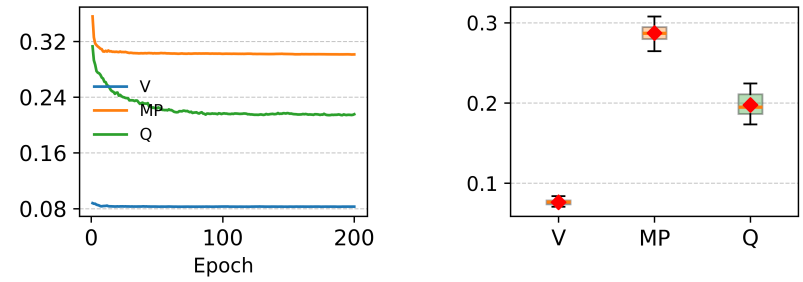
(a) CV errors for IR dataset



(b) CV errors for L



(c) CV errors for OD dataset



(c) CV errors for PBC dataset

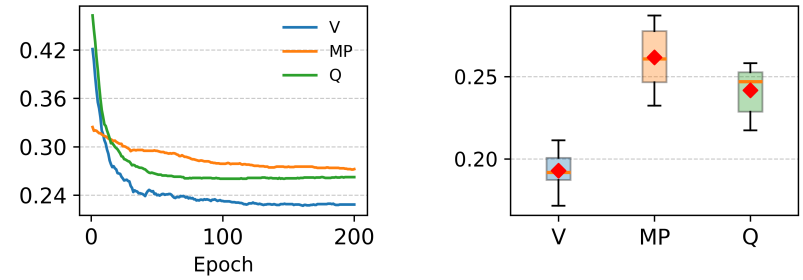
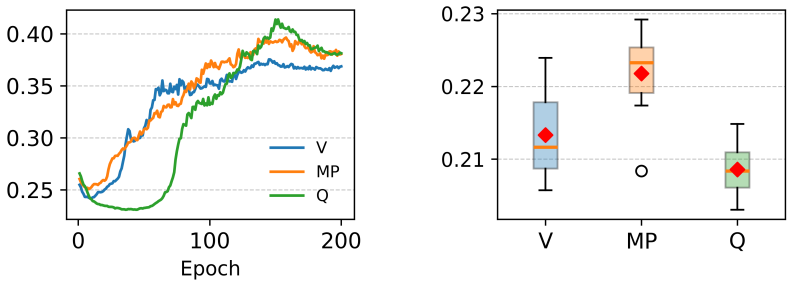
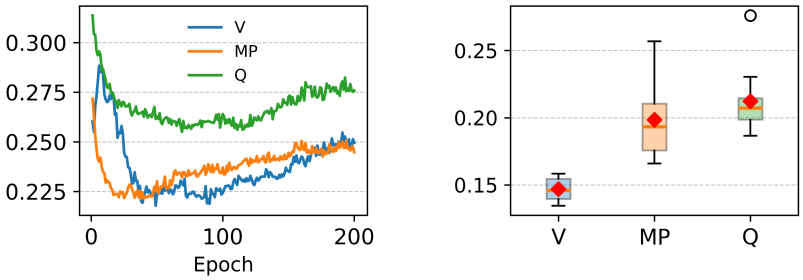


Figure 5: The trajectories of the mean cross-validation errors across epochs (left panels) and the distributions of mean minimum cross-validation errors (right panels) for the selected datasets In the boxplots, the $\overline{E}_{\text{min}}$ errors are also included (marked using red diamond).

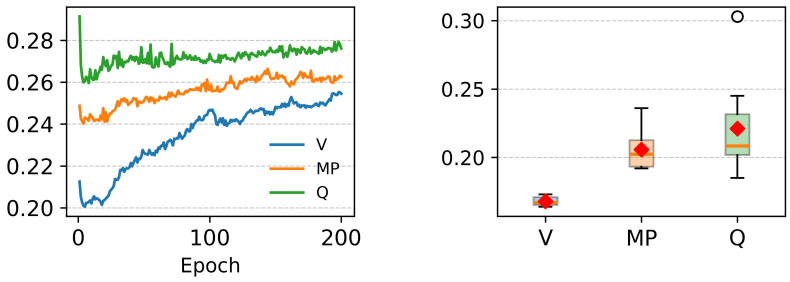
(a) CV errors for PID dataset



(b) CV errors for QSAR



(c) CV errors for SA dataset



(a) CV errors for SB dataset

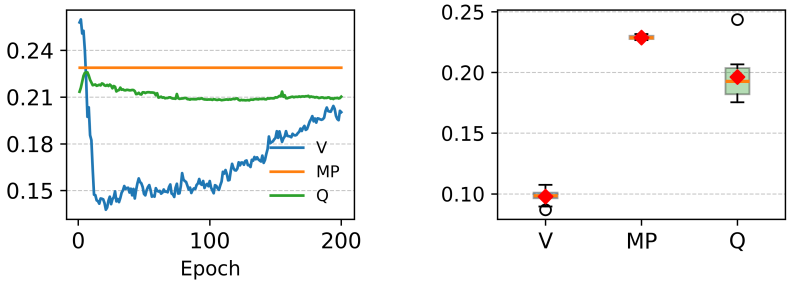


Figure 6: The trajectories of the mean cross-validation errors across epochs (left panels) and the distributions of mean minimum cross-validation errors (right panels) for the selected datasets In the boxplots, the $\overline{E}_{\text{min}}$ errors are also included (marked using red diamond).

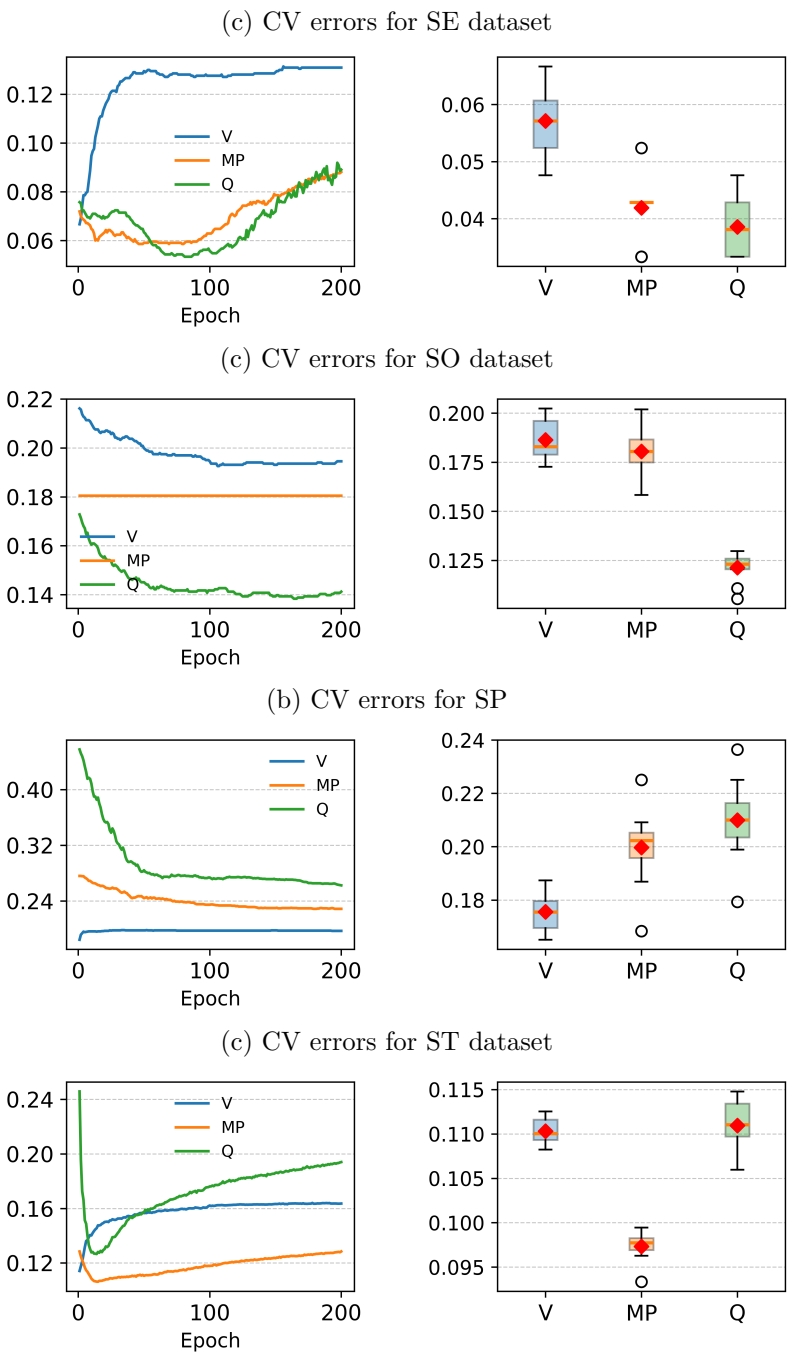


Figure 7: The trajectories of the mean cross-validation errors across epochs (left panels) and the distributions of mean minimum cross-validation errors (right panels) for the selected datasets In the boxplots, the $\overline{E}_{\text{min}}$ errors are also included (marked using red diamond).

C Comparison with published PNN results

Table 2: Descriptive comparison with published PNN results on common datasets. All values denote classification errors in percent. The results of the present study are reported in the order V/MP/Q.

Dataset	Present study: V/MP/Q	Published PNN results
D	8.20/20.10/30.70	5.73 Kusy & Zajdel (2015)
DBC	14.10/3.50/6.00	1.98 Kusy & Zajdel (2015); 4.60 for cHM Kowalski et al. (2025)
IR	3.30/2.30/2.00	0.00 on a fixed 75-sample test set Mao et al. (2000); 7.30 for cHM Kowalski et al. (2025)
L	31.00/27.90/26.50	12.79 for LSFA Alweshah & Abdullah (2015); 33.58 for GLPNN Naik et al. (2020)
OD	7.60/28.70/19.80	3.62 for the gap-based PNN evaluated on the original independent test set Zhong et al. (2007)
PID	21.30/22.20/20.90	24.71 for PCM Georgiou et al. (2006); 28.78 for B.P.EPNN Georgiou et al. (2008); 21.78 Kusy & Zajdel (2015); 9.90 for LSFA Alweshah & Abdullah (2015); 26.41 for GLPNN Naik et al. (2020); 25.20 for cHM Kowalski et al. (2025)
SO	18.60/18.10/12.10	11.14 for WLPNN Naik et al. (2020)
SP	17.60/20.00/21.00	5.97 for LSFA Alweshah & Abdullah (2015)
ST	11.00/9.70/11.10	9.70 for the gap-based PNN evaluated on the original independent test set Zhong et al. (2007)

A direct comparison with previously published PNN results is limited by differences in data partitions, validation procedures, PNN parameterizations, and performance measures. Therefore, Table 2 provides only a descriptive comparison for common datasets. All results are reported as classification errors in percent; the present study reports the V/MP/Q results, respectively.

The most closely related comparison is provided by Kusy & Zajdel (2015), which reports cross-validation results for D, DBC, and PID. The remaining studies use different evaluation protocols, including fixed train–test partitions, original independent test sets, repeated holdout procedures, alternative kernels, or optimization of PNN structure or weights. Consequently, the reported values should be interpreted as reference results rather than direct evidence of differences between training procedures.